

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号
特開2002-23622
(P2002-23622A)

(43)公開日 平成14年1月23日(2002.1.23)

(51)Int.Cl.⁷

G 0 9 C 1/00

識別記号

6 1 0

F I

G 0 9 C 1/00

テーマコード*(参考)

6 1 0 B 5 J 1 0 4

審査請求 未請求 請求項の数18 O L (全 22 頁)

(21)出願番号 特願2000-211686(P2000-211686)

(22)出願日 平成12年7月12日(2000.7.12)

(71)出願人 000003078

株式会社東芝
東京都港区芝浦一丁目1番1号

(72)発明者 村谷 博文

神奈川県川崎市幸区小向東芝町1番地 株
式会社東芝研究開発センター内

(72)発明者 本山 雅彦

神奈川県川崎市幸区小向東芝町1番地 株
式会社東芝研究開発センター内

(74)代理人 100058479

弁理士 鈴江 武彦 (外6名)

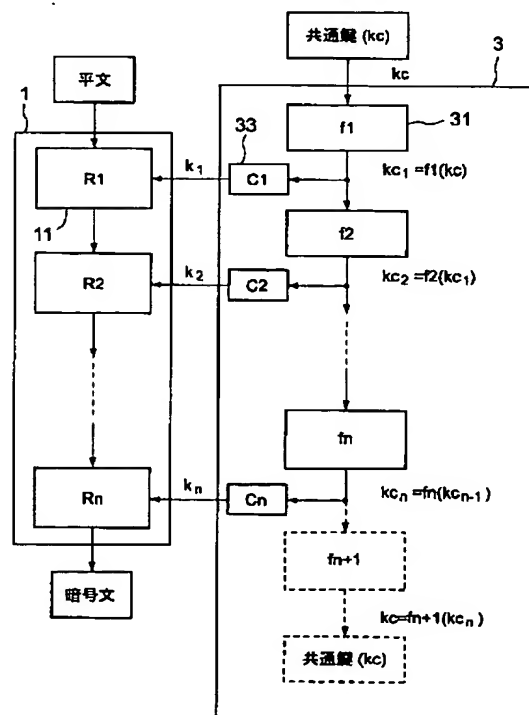
最終頁に続く

(54)【発明の名称】 暗号化装置、復号装置及び拡大鍵生成装置、拡大鍵生成方法並びに記録媒体

(57)【要約】

【課題】 拡大鍵生成のための遅延時間の発生を回避し且つOn-the-flyの鍵生成を可能とした暗号化装置を提供すること。

【解決手段】 暗号化時と復号時とで拡大鍵の使用順が逆になる共通鍵ブロック暗号方式の暗号化装置の拡大鍵生成部3において、初段からの段数と、最終段からの段数とが等しい2つのラウンド関数 f_1 と f_{n+1} とを、互いに逆関数になるように設定する。これによって、暗号化時にも復号時にも共通鍵を入力として直ちに且つ使用順に拡大鍵を逐次生成していくことが可能になる。また、暗号化時の拡大鍵生成と復号時の拡大鍵生成とが基本的に同一になる。



【特許請求の範囲】

【請求項 1】暗号化時のデータ攪拌処理と復号時のデータ攪拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、2 段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するラウンド処理手段と、

前記ラウンド処理手段の全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後前記拡大鍵として出力するための出力手段とを備え、

前記ラウンド処理手段は、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すものであることを特徴とする暗号化装置。

【請求項 2】前記ラウンド関数系列は、初段からの段数と最終段からの段数とが一致する 2 つのラウンド関数を互いに逆関数になるように設定したものであることを特徴とする請求項 1 に記載の暗号化装置。

【請求項 3】前記ラウンド関数系列は、少なくとも第 1 の特定の段と第 2 の特定の段との間の連続する複数段について、該第 1 の特定の段からの段数と該第 2 の特定の段からの段数とが一致する 2 つのラウンド関数を互いに逆関数になるように設定した部分系列を含むものであることを特徴とする請求項 1 に記載の暗号化装置。

【請求項 4】前記ラウンド関数系列は、少なくとも第 1 の特定の段から段数増加方向へ特定段数隔てた段までの連続する範囲と第 2 の特定の段から段数減少方向へ特定段数隔てた段までの連続する範囲について、該第 1 の特定の段からの段数と該第 2 の特定の段からの段数とが一致する 2 つのラウンド関数を互いに逆関数になるように設定した部分系列を含むものであることを特徴とする請求項 1 に記載の暗号化装置。

【請求項 5】前記ラウンド関数系列を、請求項 3 または 4 に記載の部分系列を含まないように設定したことを特徴とする請求項 1 に記載の暗号化装置。

【請求項 6】前記出力手段は、前記拡大鍵の出力のために前記中間状態を用いる際に、該中間状態については、その全ビットのうちから選択した当該中間状態を一意に決定するには十分ではない部分のみを用いることを特徴とする請求項 1 ないし 5 のいずれか 1 項に記載の暗号化装置。

【請求項 7】前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段若しくは初段から段数

増加方向へ所定段数隔てた段までの連続する範囲、及び又は最終段若しくは最終段から段数減少方向へ所定段数隔てた段までの連続する範囲に属するラウンド関数に対応する中間状態は、前記拡大鍵として又はそのもととなるデータとして用いないことを特徴とする請求項 1 ないし 6 のいずれか 1 項に記載の暗号化装置。

【請求項 8】前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段からの段数と最終段からの段数とが一致する 2 つのラウンド関数に対応する中間状態のいずれか一方又は両方は、前記拡大鍵として又はそのもととなるデータとして用いないことを特徴とする請求項 1 ないし 7 のいずれか 1 項に記載の暗号化装置。

【請求項 9】前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段若しくは初段から段数増加方向へ所定段数隔てた段までの連続する範囲、及び又は最終段若しくは最終段から段数減少方向へ所定段数隔てた段までの連続する範囲に属するラウンド関数に対応する中間状態のうち、初段からの段数と最終段からの段数とが一致する 2 つのラウンド関数のいずれか一方又は両方に対応する中間状態は、前記拡大鍵として又はそのもととなるデータとして用いないことを特徴とする請求項 1 ないし 6 のいずれか 1 項に記載の暗号化装置。

【請求項 10】複数の前記拡大鍵のうちの任意のものが常には一致しないようにしたことを特徴とする請求項 1 ないし 9 のいずれか 1 項に記載の暗号化装置。

【請求項 11】複数の前記拡大鍵のうちの任意のものが、それら拡大鍵の全ビットのうちの任意のビット群についても、常には一致しないようにしたことを特徴とする請求項 10 に記載の暗号化装置。

【請求項 12】前記ラウンド処理手段及び前記出力手段は、前記データ攪拌処理に必要な拡大鍵数を越える数の拡大鍵を、該データ攪拌処理に提供可能であり、前記提供可能な拡大鍵のうち実際に前記データ攪拌処理に提供すべき拡大鍵を示す情報、又は前記データ攪拌処理に提供すべき拡大鍵及びその提供する順番を示す情報を拡張共通鍵とし、

前記出力手段は、前記拡張共通鍵に従って、前記拡大鍵を出力することを特徴とする請求項 1 ないし 11 のいずれか 1 項に記載の暗号化装置。

【請求項 13】暗号化時のデータ攪拌処理と復号時のデータ攪拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による復号装置であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、2 段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するラウンド処理手段と、

前記ラウンド処理手段の全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット

3

又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するための出力手段とを備え、

前記ラウンド処理手段は、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すものであることを特徴とする復号装置。

【請求項 14】暗号化時のデータ攪拌処理と復号時のデータ攪拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置又は復号装置に用いられる拡大鍵生成装置であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、2 段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するラウンド処理手段と、

前記ラウンド処理手段の全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するための出力手段とを備え、

前記ラウンド処理手段は、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すものであることを特徴とする拡大鍵生成装置。

【請求項 15】暗号化時のデータ攪拌処理と復号時のデータ攪拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置のための拡大鍵生成方法であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、2 段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するとともに、

全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するステップを有し、

前記中間状態の生成にあたっては、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の

4

順番に従って施すことを特徴とする拡大鍵生成方法。

【請求項 16】暗号化時のデータ攪拌処理と復号時のデータ攪拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による復号装置のための拡大鍵生成方法であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、2 段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するとともに、

全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するステップを有し、

前記中間状態の生成にあたっては、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すことを特徴とする拡大鍵生成方法。

【請求項 17】暗号化時のデータ攪拌処理と復号時のデータ攪拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置のための拡大鍵生成プログラムを記録したコンピュータ読取り可能な記録媒体であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、2 段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するとともに、全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力させ、

前記中間状態の生成にあたっては、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施させるためのプログラムを記録したコンピュータ読取り可能な記録媒体。

【請求項 18】暗号化時のデータ攪拌処理と復号時のデータ攪拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による復号装置のための拡大鍵生成プログラムを記録したコンピュータ読取り可能な記録媒体であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、2 段目以降では、前段にて生成された中間状態を入

10

20

30

40

50

力として所定のラウンド関数を施して新たな中間状態を生成するとともに、全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後、前記拡大鍵として出力させ、

前記中間状態の生成にあたっては、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施させるためのプログラムを記録したコンピュータ読取り可能な記録媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、暗号化時と復号時とで複数の拡大鍵を逆の順番で用いる暗号化装置、復号装置及び拡大鍵生成装置、拡大鍵生成方法並びに記録媒体に関する。

【0002】

【従来の技術】電子化された情報、とりわけ著作権に係る情報や機密情報やプライバシーに係る情報等のセキュリティ・コントロールのために、暗号技術の重要性が非常に高くなっている。実際に暗号技術は様々な分野において様々な形で利用されている。

【0003】暗号方式には様々なものがあるが、そのうちの一つに、共通鍵暗号方式がある。共通鍵暗号方式は、暗号化の際に用いられた鍵と同一の鍵（共通鍵、秘密鍵）を用いて復号が行われる方式である。

【0004】共通鍵暗号方式にも種々のものがあるが、そのうちの一つに、拡大鍵を用いる方式がある。この方式では、共通鍵をもとに、それが持つビット数より多い総ビット数の複数の拡大鍵を生成する。

【0005】拡大鍵の生成方式の一つに、共通鍵に対してラウンド関数（段関数）を作用させ、その出力値をもとに拡大鍵を生成するとともに、さらに該出力値にラウンド関数を作用させ、その出力値をもとに次の拡大鍵を生成するとともに、さらに該出力値にラウンド関数を作用させ…、というように、次々とラウンド関数を作用させて拡大鍵を逐次的に生成していくものがある。このような方式をここではラウンド方式と呼ぶものとする。

【0006】なお、このような拡大鍵生成方式をとる共通鍵暗号方式としては、例えば、共通鍵ブロック暗号方式がある。なお、共通鍵ブロック暗号方式は、データ攪拌部についても、処理単位となる所定ビット長のブロック・データに、ラウンド関数を次々と作用させて暗号化または復号を行う構造を有するものであり、その代表的な基本構造にSPN型とFeistel型等がある。

【0007】さて、拡大鍵の生成にラウンド方式をとる

場合には、例えばブロック暗号のように、暗号化の際に用いられた順番とは逆の順番で拡大鍵を用いることが要求される。

【0008】以下、このような方式の問題点について説明する。

【0009】図38に、従来の暗号化装置の拡大鍵生成部の構成例を示す。

【0010】まず、データ攪拌部では暗号化処理に拡大鍵（1）を必要とする。そこで、共通鍵にラウンド関数（1）を作用させ、その出力値を求め、これに拡大鍵変換（1）を作用させて、拡大鍵（1）を得る。データ攪拌部は、この拡大鍵（1）を用いて暗号化処理を行う。

【0011】次に、データ攪拌部では暗号化処理に拡大鍵（2）を必要とする。そこで、ラウンド関数（1）の出力値にラウンド関数（2）を作用させ、その出力値を求め、これに拡大鍵変換（2）を作用させて、拡大鍵（2）を得る。データ攪拌部は、この拡大鍵（2）を用いて暗号化処理を行う。

【0012】以降、同様にして、拡大鍵生成部による拡大鍵の生成と、データ攪拌部による暗号化処理が行われる。

【0013】ここで、復号時の処理を考える。

【0014】復号時には、暗号時とは逆の順、すなわち、拡大鍵（n）→拡大鍵（1）の順番に、拡大鍵を用いる必要がある。ところが、図38と同様の構成の拡大鍵生成部を有する従来の復号装置では、拡大鍵は、拡大鍵（1）→拡大鍵（n）の順に生成されるので、例えば、データ攪拌部の処理に先立って、すべての拡大鍵を生成し、メモリに記憶しておく必要があった。

【0015】しかしながら、例えばICカードのように貧弱なハードウェア環境しかない装置では、復号に必要な拡大鍵をすべて格納するだけの記憶領域の余裕がないという問題点がある。

【0016】一方、この問題を回避するために、図39に例示するような構成が考えられる。この従来の復号装置の拡大鍵生成部の構成例では、一旦、暗号化時と同一の拡大鍵生成処理を行って最終ラウンドでラウンド関数を作用させて得られる出力値（1020）を求める。その後、あらためて、該出力値（1020）に暗号化時とは逆のラウンド方向に各ラウンド関数の逆関数を作用させて、拡大鍵（n）→拡大鍵（1）の順に、すなわちOn-the-flyに拡大鍵を生成していく。

【0017】しかしながら、最初に暗号化時と同一の拡大鍵生成処理を行う不要な時間のために、復号が開始されるまでの遅延時間が発生するという問題点があった。

【0018】

【発明が解決しようとする課題】以上説明したように、従来の技術では、拡大鍵を逆順に生成することはできないので、復号処理に先立って、全拡大鍵を生成し記憶しておく必要があるが、例えばICカードのように貧弱な

ハードウェア環境では、復号に必要な拡大鍵をすべて格納するだけの記憶領域の余裕がないという問題点があった。

【0019】また、On-the-flyの鍵生成によって、この問題を回避するためには、一旦、暗号化時と同一の拡大鍵生成処理を行って最終ラウンドでラウンド関数を作用させて得られる出力値を求めた後に、あらためて該出力値に逆のラウンド方向に各ラウンド関数の逆関数を作用させていくことが必要になるが、この場合も、復号が開始されるまでの遅延時間が避けられないという問題点があった。

【0020】本発明は、上記事情を考慮してなされたもので、拡大鍵生成のための遅延時間の発生を回避もしくは小さくし、かつ、On-the-flyの鍵生成を可能とした暗号化装置、復号装置及び拡大鍵生成装置、拡大鍵生成方法並びに記録媒体を提供することを目的とする。

【0021】

【課題を解決するための手段】本発明は、暗号化時のデータ攪拌処理と復号時のデータ攪拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置または復号装置であって、複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、2段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するラウンド処理手段と、前記ラウンド処理手段の全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するための出力手段とを備え、前記ラウンド処理手段は、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちのの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すものであることを特徴とする。

【0022】好ましくは、前記ラウンド関数系列は、初段からの段数と最終段からの段数とが一致する2つのラウンド関数を互いに逆関数になるように設定したものであるようにしてもよい。また、好ましくは、前記ラウンド関数系列は、少なくとも第1の特定の段と第2の特定の段との間の連続する複数段について、該第1の特定の段からの段数と該第2の特定の段からの段数とが一致する2つのラウンド関数を互いに逆関数になるように設定した部分系列を含むものであるようにしてもよい。また、好ましくは、前記ラウンド関数系列は、少なくとも第1の特定の段から段数増加方向へ特定段数隔てた段までの連続する範囲と第2の特定の段から段数減少方向へ特定段数隔てた段までの連続する範囲について、該第1

の特定の段からの段数と該第2の特定の段からの段数とが一致する2つのラウンド関数を互いに逆関数になるように設定した部分系列を含むものであるようにしてもよい。

【0023】好ましくは、前記出力手段は、前記拡大鍵の出力のために前記中間状態を用いる際に、該中間状態については、その全ビットのうちから選択した当該中間状態を一意に決定するには十分ではない部分のみを用いるようにしてもよい。

10 【0024】好ましくは、前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段若しくは初段から段数増加方向へ所定段数隔てた段までの連続する範囲、及び又は最終段若しくは最終段から段数減少方向へ所定段数隔てた段までの連続する範囲に属するラウンド関数に対応する中間状態は、前記拡大鍵として又はそのもととなるデータとして用いないようにしてもよい。また、好ましくは、前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段からの段数と最終段からの段数とが一致する2つのラウンド関数に対応する中間状態のいずれか一方又は両方は、前記拡大鍵として又はそのもととなるデータとして用いないようにしてもよい。また、好ましくは、前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段若しくは初段から段数増加方向へ所定段数隔てた段までの連続する範囲、及び又は最終段若しくは最終段から段数減少方向へ所定段数隔てた段までの連続する範囲に属するラウンド関数に対応する中間状態のうち、初段からの段数と最終段からの段数とが一致する2つのラウンド関数のいずれか一方又は両方に対応する中間状態は、前記拡大鍵として又はそのもととなるデータとして用いないようにしてもよい。

【0025】好ましくは、複数の前記拡大鍵のうちの任意のものが常には一致しないようにしてもよい。また、好ましくは、複数の前記拡大鍵のうちの任意のものが、それら拡大鍵の全ビットのうちの任意のビット群についても、常には一致しないようにしてもよい。

【0026】好ましくは、前記ラウンド処理手段及び前記出力手段は、前記データ攪拌処理に必要な拡大鍵数を越える数の拡大鍵を、該データ攪拌処理に提供可能であり、前記提供可能な拡大鍵のうち実際に前記データ攪拌処理に提供すべき拡大鍵を示す情報、又は前記データ攪拌処理に提供すべき拡大鍵及びその提供する順番を示す情報を拡張共通鍵とし、前記出力手段は、前記拡張共通鍵に従って、前記拡大鍵を出力するようにしてもよい。

【0027】また、本発明は、暗号化時のデータ攪拌処理と復号時のデータ攪拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置又は復号装置に用いられる拡大鍵生成装置であって、複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、2段目以

降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するラウンド処理手段と、前記ラウンド処理手段の全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するための出力手段とを備え、前記ラウンド処理手段は、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すものであることを特徴とする。

【0028】なお、暗号化装置に係る本発明は、暗号化方法、復号装置、復号方法、拡大鍵生成装置又は拡大鍵生成方法に係る発明としても成立する。また、それら発明は、コンピュータに当該発明に相当する手順を実行させるための（あるいはコンピュータを当該発明に相当する手段として機能させるための、あるいはコンピュータに当該発明に相当する機能を実現させるための）プログラムを記録したコンピュータ読取り可能な記録媒体としても成立する。

【0029】本発明によれば、拡大鍵生成のためのラウンド関数の系列を、共通鍵を入力し、共通鍵と同じ値を出力するように設定することによって、暗号化時と復号時の両方において、従来のような不必要な遅延時間や記憶容量の消費なく、共通鍵から *On-the-fly* に拡大鍵を生成することが可能になる。

【0030】

【発明の実施の形態】以下、図面を参照しながら発明の実施の形態を説明する。

【0031】本発明は、暗号化時と復号時とで逆の順番に拡大鍵を用いる共通鍵暗号方式のすべてに適用可能であるが、以下では、所定ビット長のブロック・データに対して逐次的に各拡大鍵を用いたデータ攪拌処理を行っていくような共通鍵ブロック暗号方式を例にとって説明する。

【0032】まず、本実施形態の基本的な構成例について説明する。

【0033】なお、以下で参照する各図では（当該暗号に着目して説明するため）暗号対象となるデータを平文として示してあるが、もちろん、暗号対象となるデータがすでに同一のまたは他の暗号方式によって暗号化されたものであってもよい。

【0034】また、本暗号方式は、ハードウェアによってもソフトウェアによっても実現可能であり、以下に示す構成例は、暗号化装置（復号装置）の機能ブロック図としても成立し、また暗号アルゴリズム（復号アルゴリズム）の機能モジュール図もしくはフローチャート図と

しても成立する。

【0035】図1に、本発明の一実施形態に係る暗号化装置の構成例を示す。

【0036】図1に示されるように、本暗号化装置は、データ攪拌部1と、拡大鍵生成部3を備えている。

【0037】拡大鍵生成部3は、複数のラウンド処理部31を備えている。

【0038】f1で示したラウンド処理部は、共通鍵kcにラウンド関数f1を作用させて中間状態kc1 = f1(kc)を出力する。f2で示したラウンド処理部は、前段のラウンド処理部の出力した中間状態kc1にラウンド関数f2を作用させて中間状態kc2 = f2(kc1) = f2(f1(kc))を出力する。図示しないf3～fn-1に対応するラウンド処理部も同様である。fnで示したラウンド処理部は、前段のラウンド処理部の出力した中間状態kc_{n-1}にラウンド関数fnを作用させて中間状態kc_n = fn(kc_{n-1}) = fn(fn-1(...f2(f1(kc))...))を出力する。

【0039】ここで、本実施形態では、fn+1で示したラウンド処理部にて前段のラウンド処理部の出力した中間状態kc_nにラウンド関数fn+1を作用させ、これによって得られた出力値kc_{n+1} = fn+1(kc_n) = fn+1(fn(fn-1(...f2(f1(kc))...)))が、共通鍵kcに等しくなるようする。また、このラウンド関数fn+1の逆関数fn+1^{-1}が、復号装置における拡大鍵生成部の初段のラウンド処理部のラウンド関数になる。なお、図1の構成例では、この暗号化装置における拡大鍵生成部にはラウンド関数fn+1のラウンド処理部を備えなくても構わない（備えても構わない）。

【0040】また、拡大鍵生成部3は、複数の拡大鍵変換部33を備えている。

【0041】c1で示した拡大鍵変換部は、f1で示したラウンド処理部の出力kc1の全部または一部に拡大鍵変換関数c1を作用させて、共通鍵k1を生成する。c2で示した拡大鍵変換部は、f2で示したラウンド処理部の出力kc2の全部または一部に拡大鍵変換関数c2を作用させて、共通鍵k2を生成する。図示しないc3～cn-1に対応するラウンド処理部も同様である。cnで示した拡大鍵変換部は、fnで示したラウンド処理部の出力kc_nの全部または一部に拡大鍵変換関数cnを作用させて、共通鍵knを生成する。

【0042】データ攪拌部1は、ここでは、従属接続された複数の（例えばラウンド関数による）攪拌処理部11を備えるものとしている。

【0043】R1で示した攪拌処理部は、暗号化の対象となるブロック・データを入力し、拡大鍵k1を用いて、攪拌処理R1を行う。R2で示した攪拌処理部は、R2で示した攪拌処理部から出力されたブロック・データを入力し、拡大鍵k2を用いて、攪拌処理R2を行う。図示しないR3～Rn-1に対応する攪拌処理部も同

様である。 R_n で示した搅拌処理部は、 R_{n-1} で示した搅拌処理部から出力されたブロック・データを入力し、拡大鍵 k_n を用いて、搅拌処理 R_n を行う。 R_n で示した搅拌処理部の出力が、求めるべき暗号文となる。

【0044】なお、複数のラウンド関数は、すべて異なるものであっても、すべて同じものであっても、異なるものと同じものが混在するものであってもよい。複数のラウンド関数を異なるものにする場合に、関数を異ならせる方法の他に、基本的には同じ関数であるが段に応じて異なる定数に依存するものにする方法などがある。また、複数のラウンド関数は、すべて線形関数であっても任意の関数であってもよいが、それらのうちの少なくとも一つを非線形関数にするのが好ましい。また、2以上のラウンド関数あるいは全てのラウンド関数を非線形関数にしてもよい。また、ラウンド関数は、変換テーブルを用いて実現する方法、行列演算やその他の演算で実現する方法、実回路によって実現する方法など、種々の構成方法がある。これらの点は、複数の拡大鍵変換関数についても同様である。なお、拡大鍵生成部3として、入力された中間状態またはその一部を、そのまま拡大鍵として出力する構成にする（あるいは、中間状態をデータ搅拌部1（または後述するスイッチング回路15）に直結する）ことも可能である。

【0045】なお、ブロック・データのデータ長と、共通鍵 k_c のデータ長とは、同じであってもよいし、異なるものであってもよい。また、拡大鍵のデータ長と、ブロック・データのデータ長とは、同じであってもよいし、異なるものであってもよい。また、中間状態のデータ長と、拡大鍵のデータ長とは、同じであってもよいし、異なるものであってもよい。

【0046】図2に、本発明の一実施形態に係る復号装置の構成例を示す。

【0047】図2に示されるように、本復号装置は、データ搅拌部2と、拡大鍵生成部4を備えている。図2の復号装置は、図1の暗号化装置の逆変換を行う機能を有するものである。

【0048】拡大鍵生成部4は、複数のラウンド処理部42を備えており、図1の暗号化装置の拡大鍵生成部2の複数のラウンド関数の各々の逆関数を、逆の順番で作用させるものである。

【0049】 f_{n+1}^{-1} で示したラウンド処理部は、共通鍵 $k_c = f_{n+1}(k_{cn}) = f_{n+1}(f_n(f_{n-1}(\dots f_2(f_1(k_c))\dots)))$ にラウンド関数 f_{n+1}^{-1} を作用させて、中間状態 $k_{cn} = f_{n+1}^{-1}(k_c) = f_{n+1}^{-1}(f_{n+1}(f_n(f_{n-1}(\dots f_2(f_1(k_c))\dots))) = f_n(f_{n-1}(\dots f_2(f_1(k_c))\dots))$ を出力する。 f_n^{-1} で示したラウンド処理部は、前段のラウンド処理部の出力した中間状態 k_{cn} にラウンド関数 f_{n+1}^{-1} を作用させて、 $k_{cn-1} = f_{n-1}(\dots f_2(f_1(k_c))\dots)$ を出力する。図示しな

い $f_{n-1} \sim f_3$ に対応するラウンド処理部も同様である。 f_2^{-1} で示したラウンド処理部は、前段のラウンド処理部の出力した中間状態 $k_{c2} = f_2(f_1(k_c))$ にラウンド関数 f_2^{-1} を作用させて、 $k_{c1} = f_1(k_c)$ を出力する。

【0050】ここで、本実施形態では、 f_1 で示したラウンド処理部にて前段のラウンド処理部の出力した中間状態 k_{c2} にラウンド関数 f_1^{-1} を作用させ、これによって得られた出力値は共通鍵 k_c に等しくなる。また、このラウンド関数 f_1^{-1} の逆関数 f_1 が、暗号化装置における拡大鍵生成部の初段のラウンド処理部のラウンド関数になる。なお、図2の構成例では、この復号装置には拡大鍵生成部ではラウンド関数 f_1^{-1} のラウンド処理部を備えなくても構わない（備えても構わない）。

【0051】また、拡大鍵生成部3は、複数の拡大鍵変換部44を備えている。この部分は、図1の暗号化装置の対応する部分と同じ処理内容となる。

【0052】データ搅拌部2は、ここでは、従属接続された複数の（例えばラウンド関数による）搅拌処理部22を備えるものとしている。 R_n^{-1} で示した搅拌処理部は、復号の対象となるブロック・データを入力し、拡大鍵 k_n を用いて、暗号化装置の搅拌処理 R_n の逆変換となる搅拌処理 R_n^{-1} を行う。同様に、他の搅拌処理部も順次、前段の搅拌処理部から出力されるブロック・データを入力し、拡大鍵 k_{n-1} 、 \dots 、または k_2 を用いて、搅拌処理 R_{n-1}^{-1} 、 \dots 、または R_2^{-1} を行う。 R_1^{-1} で示した搅拌処理部は、 R_2^{-1} で示した搅拌処理部から出力されたブロック・データを入力し、拡大鍵 k_1 を用いて、暗号化装置の搅拌処理 R_1 の逆変換となる搅拌処理 R_1^{-1} を行う。 R_1^{-1} で示した搅拌処理部の出力によって、求めるべき復号結果となるブロック・データが与えられる。

【0053】すなわち、復号時において、すぐに暗号化時とは逆の順番での拡大鍵の生成に入り、拡大鍵を次々と生成していくことができる。

【0054】以上のように、暗号化時のラウンド関数の系列（ただし、最終段は備えないこともある）と、その逆関数となる復号時のラウンド関数の系列（ただし、最終段は備えないこともある）とについて、暗号化時の最終段の出力に相当する値がもとの共通鍵に一致するようにラウンド関数の系列を設定することによって、暗号化時と復号時の両方において、従来のような不必要な遅延時間や記憶容量の消費なく、共通鍵からOn-the-flyに拡大鍵を生成することが可能になる。

【0055】次に、図1／図2の暗号化装置や復号装置の拡大鍵生成部の複数のラウンド処理部によるラウンド関数の系列について説明する。なお、暗号化装置におけるラウンド関数の系列と、復号装置におけるラウンド関数の系列とは、逆関数の関係になるので、一方が決まれば、他方も決まることになる。ここでは、暗号化装置の

方を例にとつて説明する。

【0056】ラウンド関数の系列 f_1, f_2, \dots, f_{n+1} について、そのラウンド関数の系列の内容、あるいは各々の順番のラウンド関数の内容は、当該ラウンド関数の系列が全体として共通鍵を入力し共通鍵と同じ値を出力する条件を満たす範囲内において適宜設定可能であり、様々なバリエーションがある。以下、ラウンド関数の系列のバリエーションのいくつかを例示列挙的に説明する。

【0057】（ラウンド・トリップ構成）まず、ラウンド・トリップ構成について説明する。

【0058】ここでは、ラウンド関数の系列の段数を $2r$ 段とする（なお、前述したように、 $2r$ 段目のラウンド関数は、備えられないことがある）。

【0059】ラウンド関数の系列を構成する一つの方法は、 $0 \leq i \leq r$ を満たすすべての i について、第 $(r+i)$ 段目の段関数を、第 $(r-i+1)$ 段目の段関数の逆関数になっている、という関係を満たすように構成する方法である。

【0060】例えば、ラウンド関数の系列を 8 段、すな

わち、
 $f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8$
 とし、 $f_1 \sim f_4$ を任意のラウンド関数として、 $f_5 = f_4^{-1}$ 、 $f_6 = f_3^{-1}$ 、 $f_7 = f_2^{-1}$ 、 $f_8 = f_1^{-1}$ とすると、

$f_1, f_2, f_3, f_4, f_4^{-1}, f_3^{-1}, f_2^{-1}, f_1^{-1}$

という順番の系列となる。すなわち、共通鍵を入力として、 $f_1, f_2, f_3, f_4, f_4^{-1}, f_3^{-1}, f_2^{-1}, f_1^{-1}$ を次々と作用させることによって、最終段の出力が共通鍵と一致するようになる。

【0061】このような構成を、ラウンド・トリップ構成と呼ぶものとする。また、この様子を図 3 に概念的に示す。

【0062】ラウンド・トリップ構成を採用した場合、暗号化装置におけるラウンド関数の系列と、復号装置におけるラウンド関数の系列とが同一になる。

【0063】上記の例の場合、暗号化装置における 8 段のラウンド関数を、

$f_1, f_2, f_3, f_4, f_4^{-1}, f_3^{-1}, f_2^{-1}, f_1^{-1}$

とすると、復号装置における 8 段のラウンド関数は、この逆関数となるので、

$(f_1^{-1})^{-1}, (f_2^{-1})^{-1}, (f_3^{-1})^{-1}, (f_4^{-1})^{-1}, (f_4)^{-1}, (f_3)^{-1}, (f_2)^{-1}, (f_1)^{-1}$

であり、したがって、

$f_1, f_2, f_3, f_4, f_4^{-1}, f_3^{-1}, f_2^{-1}, f_1^{-1}$

となり、両者が一致することがわかる。

【0064】なお、暗号化装置においても最終段のラウンド関数（上記の例では、 f_1^{-1} ）を備えなくてもよいし、復号装置においても最終段のラウンド関数（上記の例では、 f_1 ）を備えなくてもよいが、いずれも最終段のラウンド関数を備えれば、構成が同一となるので、暗号化機能と復号機能の両方の機能を兼ね備えさせる装置においては、暗号化時と復号時とで 1 つの拡大鍵生成部を兼用することによって、装置規模を削減することも可能である。

【0065】次に、この構成において、ラウンド関数の系列の前半の各ラウンド関数は、すべて異なるものであっても、すべて同じものであっても、異なるものと同じものが混在するものであってもよい。

【0066】例えば、ラウンド関数の系列の前半の各ラウンド関数がすべて同じものである場合、段数を 8 段とすると、暗号側と復号側のいずれにおいても、

$f_1, f_1, f_1, f_1, f_1^{-1}, f_1^{-1}, f_1^{-1}, f_1^{-1}$

という系列になる。

【0067】ところで、ラウンド・トリップ構成を採用した場合、ラウンド関数の系列において逆関数の関係にある対応部分の中間状態が同一になる。したがって、同一の中間状態に同一の拡大鍵変換関数を作用させると、同一の拡大鍵が生成される。そこで、これを避けるために、ラウンド関数の系列において逆関数の関係にある対応部分についての 2 つの拡大鍵変換部の拡大鍵変換関数として相異なるものを用いるようにしてもよい。

【0068】例えば、8 段のラウンド関数の系列を、
 $f_1, f_2, f_3, f_4, f_4^{-1}, f_3^{-1}, f_2^{-1}, f_1^{-1}$

とし、 f_1 の出力を用いる拡大鍵変換関数を c_1 、 \dots 、 f_2^{-1} の出力を用いる拡大鍵変換関数を c_7 とすると、拡大鍵変換関数 c_1 と拡大鍵変換関数 c_7 とを異ならせるようにしてもよい。 c_2 と c_6 、 c_3 と c_5 についても同様である。

【0069】（ループ構成）次に、ループ構成について説明する。

【0070】ラウンド・トリップ構成では、ラウンド関数の系列の後半を、その前半の逆関数としたが、ラウンド関数の系列の中の部分系列として、ラウンド・トリップ構成に相当する部分を全く持たない構成も可能である。

【0071】このような構成を、ループ構成と呼ぶものとする。また、この様子を図 4 に概念的に示す。

【0072】なお、ラウンド・トリップ構成では、ラウンド関数の系列の段数を偶数段としたが、ループ構成では、ラウンド関数の系列の段数は、偶数段であっても奇数段であってもよい。

【0073】例えば、ラウンド関数の系列を 8 段とした場合に、共通鍵を入力として、 f_1, f_2, f_3, f

10

20

30

40

50

4、 f_5 、 f_6 、 f_7 、 f_8 を次々と作用させることによって、最終段の出力が共通鍵と一致するようにする。この場合、その逆関数は、

f_8^{-1} 、 f_7^{-1} 、 f_6^{-1} 、 f_5^{-1} 、 f_4^{-1} 、 f_3^{-1} 、 f_2^{-1} 、 f_1^{-1}

であり、共通鍵を入力すると、最終段の出力が共通鍵と一致することになる。

【0074】また、例えば、ラウンド関数の系列の前半の各ラウンド関数がすべて同じものである場合、段数を8段とすると、暗号側では、

f_1 、 f_1 、 f_1 、 f_1 、 f_1 、 f_1 、 f_1 、 f_1

という系列になり、復号側では、

f_1^{-1} 、 f_1^{-1} 、 f_1^{-1} 、 f_1^{-1} 、 f_1^{-1} 、 f_1^{-1} 、 f_1^{-1} 、 f_1^{-1}

という系列になる。

【0075】なお、このような条件を満たす関数は、シフト演算、行列演算、ガロア体演算など、種々のものがある。

【0076】(ラウンド・トリップ／ループ複合構成) ラウンド関数の系列としては、その部分系列として、ラウンド・トリップ構成に相当する部分と、ループ構成に相当する部分とを複合的に備えるような構成も可能である。

【0077】以下、ラウンド・トリップ構成部分を図3の表記方法で示し、ループ構成部分を図3の表記方法で示すものとして、図5～図9にいくつかのバリエーションを例示する。

【0078】図5の例は、ラウンド・トリップ構成部分の途中にラウンド・トリップ構成部分を含むような入れ子構造になっているものである。図5の場合のラウンド関数の系列を例示すると、

$a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow b_1 \rightarrow b_2 \rightarrow b_2^{-1} \rightarrow b_1^{-1} \rightarrow a_4 \rightarrow a_5 \rightarrow a_6 \rightarrow a_6^{-1} \rightarrow a_5^{-1} \rightarrow a_4^{-1} \rightarrow a_3^{-1} \rightarrow c_1 \rightarrow c_2 \rightarrow c_2^{-1} \rightarrow d_1 \rightarrow d_1^{-1} \rightarrow c_1^{-1} \rightarrow a_2^{-1} \rightarrow a_1^{-1}$

となる。この例の場合、 $a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_4 \rightarrow a_5 \rightarrow a_6 \rightarrow a_6^{-1} \rightarrow a_5^{-1} \rightarrow a_4^{-1} \rightarrow a_3^{-1} \rightarrow a_2^{-1} \rightarrow a_1^{-1}$ というラウンド・トリップ構成の中に、 $b_1 \rightarrow b_2 \rightarrow b_2^{-1} \rightarrow b_1^{-1}$ というラウンド・トリップ構成と、 $c_1 \rightarrow c_2 \rightarrow c_2^{-1} \rightarrow c_1^{-1}$ というラウンド・トリップ構成が入っており、さらに、 $c_1 \rightarrow c_2 \rightarrow c_2^{-1} \rightarrow c_1^{-1}$ というラウンド・トリップ構成の中に、 $d_1 \rightarrow d_1^{-1}$ というラウンド・トリップ構成が入っている。

【0079】図6の例は、ループ構成部分の途中にループ構成部分を含むような入れ子構造になっているものである。図6の場合のラウンド関数の系列を例示すると、

$s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_4 \rightarrow t_1 \rightarrow t_2 \rightarrow t_3 \rightarrow s_5 \rightarrow s_6 \rightarrow s_7 \rightarrow s_8$

となる。この例の場合、 $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_4 \rightarrow s_5 \rightarrow s_6 \rightarrow s_7 \rightarrow s_8$ というループ構成の中に、 $t_1 \rightarrow t$

$2 \rightarrow t_3$ というループ構成が入っている。

【0080】図7の例は、ループ構成部分の途中にラウンド・トリップ構成部分を含むような構造になっているものである。図7の場合のラウンド関数の系列を例示すると、

$s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_4 \rightarrow s_5 \rightarrow a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_3^{-1} \rightarrow a_2^{-1} \rightarrow a_1^{-1} \rightarrow s_6 \rightarrow s_7 \rightarrow s_8 \rightarrow s_9$

となる。この例の場合、 $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_4 \rightarrow s_5 \rightarrow s_6 \rightarrow s_7 \rightarrow s_8 \rightarrow s_9$ というループ構成の中に、 $a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_3^{-1} \rightarrow a_2^{-1} \rightarrow a_1^{-1}$ というラウンド・トリップ構成が入っている。

【0081】図8の例は、ラウンド・トリップ構成部分の途中にループ構成部分を含むような構造になっているものである。図8の場合のラウンド関数の系列を例示すると、

$a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_4 \rightarrow a_5 \rightarrow a_6 \rightarrow s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_4 \rightarrow a_6^{-1} \rightarrow a_5^{-1} \rightarrow a_4^{-1} \rightarrow a_3^{-1} \rightarrow a_2^{-1} \rightarrow a_1^{-1}$

となる。この例の場合、 $a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow a_4 \rightarrow a_5 \rightarrow a_6 \rightarrow a_6^{-1} \rightarrow a_5^{-1} \rightarrow a_4^{-1} \rightarrow a_3^{-1} \rightarrow a_2^{-1} \rightarrow a_1^{-1}$ というラウンド・トリップ構成の中に、 $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_4$ というループ構成が入っている。

【0082】図9の例は、4つのラウンド・トリップ構成部分と、2つのループ部分を持つものである。

【0083】もちろん、これら以外にも、ラウンド・トリップ構成部分とループ構成部分の組み合わせ方、あるいは階層構造の取り方など、種々のバリエーションが可能である。

【0084】さて、図1／図2の構成例は、データ攪拌部で必要とする数の分だけ拡大鍵を生成することを想定したものであったが、データ攪拌部で必要とする数を越える数の分の拡大鍵を生成可能とするラウンド関数の段数を備え、生成可能な拡大鍵の一部をデータ攪拌部で使用する構成も可能である。

【0085】この場合の図1／図2の暗号化装置／復号装置に対応する構成例を、図10／図11にそれぞれ示す。

【0086】ここでは、図1／図2と相違する点を中心に説明する。もちろん、ラウンド関数の系列は、上記したラウンド・トリップ構成等を採用したものであってよい。

【0087】図10の5と図11の6は、拡大鍵 k_i と攪拌処理 R_j との接続関係を示す部分であり、そのいくつかの具体例を図12～図15に例示している。なお、本実施形態では、図10の5の接続関係と図11の6の接続関係は同一になる。

【0088】なお、ここでは、初段のラウンド関数への入力としての共通鍵と、最終段のラウンド関数からの出力されることになる共通鍵の双方または一方をも、中間状態として拡大鍵生成に利用してもよい。なお、後者を

10

20

30

40

50

用いる場合に、実際に最終段のラウンド関数の出力を用いるようにしてもよいし、共通鍵を記憶しておいてこれを用いるようにしてもよい。

【0089】本実施形態では、生成可能な拡大鍵の数が、攪拌処理に必要な拡大鍵の数より多くなるように構成し、拡大鍵 k_i と攪拌処理 R_j とを適宜対応付ける。なお、同一の拡大鍵を複数の攪拌処理に使用可能とする方法と、1つの拡大鍵を攪拌処理について排他的に使用可能とする方法とがある。

【0090】なお、使用しないことになる拡大鍵は、生成しないようにして構わない。この場合には、対応する拡大鍵変換部を備えなくて構わない。

【0091】このように生成しうる拡大鍵の一部のみをデータ攪拌に使用する構成は、攻撃に対する安全性の面で効果的である。

【0092】以下、種々のバリエーションについて説明する。

【0093】まず、データ攪拌部の攪拌処理の段数が n であり、(拡大鍵変換部を備えたとして)生成可能な拡大鍵の個数が m ($m > n$)である場合に、基本的には、拡大鍵の重複使用を許さない構成では、 m 個の拡大鍵のうちから n 個の拡大鍵を任意に選択したすべての組み合わせが可能である。なお、ここでは、生成される順番に拡大鍵を使用するものとする。

【0094】なお、拡大鍵の重複使用を許す構成では、基本的には、 n^m 通りの組み合わせが可能である。

【0095】いずれの拡大鍵を選択するかについては、ランダムに選択する方法の他に、所定の基準に従って選択する方法がある。

【0096】SQUARE攻撃と呼ばれる特殊な攻撃では、従来の暗号方式に対して、初段(あるいは最初からの連続する数段)あるいは最終段(あるいは最終段までの連続する数段)の拡大鍵のうち一部のビットに対して、全数探索が行われる。この場合、初段と最終段の拡大鍵が同一である場合には、探索の空間が小さくなってしまい、解読される可能性が高くなる。

【0097】そこで、初段の拡大鍵変換部により得られる拡大鍵(以下、初段の拡大鍵と呼ぶ)と最終段の拡大鍵変換部により得られる拡大鍵(以下、最終段の拡大鍵と呼ぶ)については、高々一方のみをデータ攪拌に使用するようにしてもよい(いずれか一方のみをデータ攪拌に使用する方法と、いずれもデータ攪拌に使用しない方法とがある)。

【0098】また、同様に、初段からの連続する数段分の拡大鍵と最終段までの連続する数段分の拡大鍵の範囲において、初段または最終段からの段数を同じくする2つの段の拡大鍵の組について、いずれの組においても、高々一方のみをデータ攪拌に使用するようにしてもよい。この場合に、使用するものあるいは使用しないものについての選択方法には種々のバリエーションが考えら

れる。例えば、いずれの組においても、いずれか一方を使用するものとする場合に、各組ごとに、使用する方

(または使用しない方)を、ランダムに選択してもよいし、例えば前半と後半とから交互になるように選択するなど一定の基準に従って選択するようにしてもよい。また、例えば、各組ごとに、段の順番が前側のものを使用するか、段の順番が後ろ側のものを使用するか、いずれも使用しないかを、ランダムに選択してもよいし、一定の基準に従って選択するようにしてもよい。例えば、図12に示すように、拡大鍵が15段分生成可能であり、攪拌処理が9段ある場合に、初段の拡大鍵 k_1 と最終段の拡大鍵 k_{15} のうちでは k_{15} を使用し、その1つ内側の段の k_2 と k_{14} のうちからは k_2 を使用し、同様に、 k_3 と k_{13} のうちからは k_{13} を使用し、 k_4 と k_{12} のうちからは k_4 を使用し、 k_5 と k_{11} のうちからは k_{11} を使用し、 k_6 と k_{10} のうちからは k_6 を使用するようにしてもよい。なお、この場合においても、生成される順番に拡大鍵を使用するようにしている。

【0099】また、初段からの連続する数段分の拡大鍵と最終段までの連続する数段分の拡大鍵の範囲については使用しないようにしてもよい。図13に、このときの様子を示す。

【0100】また、初段と最終段、または初段からの連続する数段分の拡大鍵と最終段までの連続する数段分の拡大鍵の範囲については使用せず、その内側の連続する数段分の拡大鍵の範囲については前述と同様に対応する1組のうち的一方のみを使用するようにしてもよい。図14に、このときの様子を示す。

【0101】もちろん、以上の例の他にも、種々のバリエーションがある。

【0102】ところで、これまでは、拡大鍵を生成可能な順番で、データ攪拌に使用するものとして説明したが、メモリ等のハードウェアまたは計算時間にある程度の余裕があれば、その余裕に応じて、拡大鍵が生成される順番と、拡大鍵をデータ攪拌に使用する順番とを入れ替えるようにしてもよい。この順番の入れ替えは、図1や図2の構成においても同様である。この順番の入れ替えも、攻撃に対する安全性の面で効果的である。

【0103】図15に、拡大鍵が生成される順番と、拡大鍵をデータ攪拌に使用する順番とを入れ替えた例を示す。

【0104】なお、順番を入れ替える場合には、例えば、先に生成された拡大鍵を、後に生成された拡大鍵よりも後で使用するために、一旦、メモリに格納しておくようにすればよい。1つの拡大鍵の順番を入れ替えるだけならば、1つの拡大鍵を一時保存するのに必要なメモリ容量が増加するだけである。

【0105】メモリを増加させないためには、中間状態にラウンド関数の逆関数を作動させて必要な中間状態を

10

20

30

40

50

復元させればよい。例えば、初段のラウンド関数 f_1 による中間状態 k_{c1} から得られる拡大鍵 k_1 を、2 段目のラウンド関数 f_2 による中間状態 k_{c2} から得られる拡大鍵 k_2 を使用した後に使用する場合、一旦、中間状態 k_{c2} を求めた後に、 k_{c2} に 2 段目のラウンド関数 f_2 の逆関数 f_2^{-1} を作用させて中間状態 k_{c1} を求め

(これによって拡大鍵 k_1 が得られる)、さらに中間状態 k_{c1} に 2 段目のラウンド関数 f_2 を作用させて中間状態 k_{c2} を求め、これに 3 段目のラウンド関数 f_3 を作用させる…、というようにすることによって、使用順に拡大鍵を生成することができる。なお、ラウンド関数系列がラウンド・トリップ構成を有する場合には、ラウンド関数 f_2 の逆関数 f_2^{-1} も同時に備えているので、これを上記の処理に利用するようにしてもよい。

【0106】ところで、以上の拡大鍵の選択や順序の入れ替えは、固定的なものであったが、これを可変とするようにしてもよい。

【0107】この場合の図 10 / 図 11 の暗号化装置 / 復号装置に対応する構成例を、図 16 / 図 17 にそれぞれ示す。図中の 7 と 8 はデコーダであり、15 と 16 は

スイッチング回路である。

【0108】この場合、予め各々の攪拌処理 R_j へ拡大鍵 k_i を対応付ける接続パターン (図 12 等参照) を複数種類用意しておき、各パターンをコード化し、これを拡張共通鍵 $k_{c'}$ として、本来の共通鍵 k_c に付加する。

【0109】暗号化時には、拡張共通鍵 $k_{c'}$ は、デコーダ 7 に与えられ、デコーダ 7 は、拡張共通鍵 $k_{c'}$ を解読し、該拡張共通鍵 $k_{c'}$ が示す接続パターン (例えば、図 12 等のパターン) を実現するようにスイッチング回路 15 に対するスイッチング制御を行う。これらの動作は、復号時と同様である。

【0110】なお、上記では、パターンをコード化するようにしたが、その代わりに、使用しない拡大鍵の段数を示す情報など、他の形態の情報を用いることも可能である。

【0111】このような構成も、攻撃に対する安全性の面で効果的である。

【0112】なお、以上の各構成例において、図 18 に示すように、初段および最終段に擬アダマール変換のような補助関数を挿入するようにしてもよい。この場合に、初段の補助関数と最終段の補助関数に同一の拡大鍵 (例えば、初段の拡大鍵) を用いるようにしてもよい。なお、擬アダマール変換は、例えば、ブロック・データの左半分と右半分の算術加算を取ったものを新たな左半分とし、この新たな左半分と右半分の算術加算を取ったものを新たな右半分とするような処理が、これに該当する。

【0113】以下では、暗号化装置や復号装置の拡大鍵生成部の複数の拡大鍵変換部のバリエーションについて

説明する。

【0114】図 19 には、1 つの拡大鍵生成部の構成例を示す。図 19 において、101 は 8 ビットの S-box であり、103 は MDS (Maximum Distance Separable; 最大距離分離) 行列に基づく 32 ビットの攪拌部である。この例は、中間状態の全部または一部として $32 \times k$ ビットのデータを入力し、 $32 \times k$ ビットの拡大鍵を出力するもので、4 並列の S-box 101 に攪拌部 103 を接続したものを 1 単位として、これを k 個分並列に設けたものである。

【0115】もちろん、前述したように、拡大鍵生成部は、種々の構成が可能である。

【0116】ところで、ある種の暗号解読によりある段 (一般には最終段) の拡大鍵 (の一部) が知られるおそれは完全には否定できない。かりに、ある段の拡大鍵が知られてしまった場合、拡大鍵変換部の逆変換を行うことで、その段の (ラウンド関数についての) 中間状態を知られてしまい、その結果として、他の段の中間状態すべてが知られるところとなり、結局、すべての拡大鍵が知られるところとなるおそれがある。そこで、一部 (例えば最終段を含む 1 段または数段) の拡大鍵変換部においては、逆変換が容易でない関数 (例えば、べき乗関数) や、逆が一意に定まらない関数 (例えば、多対一関数) を用いるようにしてもよい。これによって、他の段の拡大鍵を容易には知られないようにすることが可能になり、安全性を維持することができる。もちろん、全部の拡大鍵変換部について、逆変換が容易でない関数や、逆が一意に定まらない関数を用いるようにしてもよい。

【0117】また、拡大鍵変換部には、対応する段の中間状態の全データを与えてもよいが、その代わりに、対応する段の中間状態の一部のみを渡す構成にして、中間状態の全データを知られないようにし、これによって安全性を維持することもできる。

【0118】また、サイドチャネル解析と呼ばれる特殊な攻撃では、ハードウェアで構成された暗号化装置に対して、電力、電磁波等、IC カード等の装置から漏洩する情報をもとに鍵の推測を行う。特に、データ攪拌処理におけるある回路において、同一の構成を有する複数の回路部分があり、それらの回路への入力ビット列とその回路で使用される鍵ビット列 (拡大鍵自体または拡大鍵の一部のデータ) が同一であったならば、サイドチャネル情報 (例えば、消費電流の変化) の同一性から、それら回路について、入力されたビット列が同一であったことが推測されてしまう。したがって、拡大鍵生成においては、IC カード等で問題となるサイドチャネル解析が容易となるような鍵を生成しないものが望まれる。

【0119】そこで、少なくとも処理要素 (回路部分) の入出力の一部が直接観測あるいは推定が可能な異なる処理要素 (回路部分) において、同一の拡大鍵が使われないような拡大鍵生成方法が有効である。

10

20

30

40

50

【0120】なお、全拡大鍵が常には一致しないように、拡大鍵変換部、または拡大鍵変換部およびラウンド処理部、またはラウンド処理部を設計し、偶然一致することは許すようにしてもよい。

【0121】また、全拡大鍵が常には一致しないように、拡大鍵変換部、または拡大鍵変換部およびラウンド処理部、またはラウンド処理部を設計するとともに、共通鍵の生成時に、全拡大鍵が異なるかどうかをチェックし、全拡大鍵が異なると判定された場合にのみ、その共通鍵を使用するようにしてもよい。

【0122】ここで、拡大鍵の一致については、様々なレベルが考えられる。例えば、2つの拡大鍵の全ビットが同一のときに、一致したと判断するようにしてもよい。また、2つの拡大鍵の特定のバイト位置のデータが同一のときには、一致したと判断するようにしてもよい。また、2つの拡大鍵の全ビット、または2つの拡大鍵の特定のバイト位置のデータの間に、一定の関係があるときには、一致したと判断するようにしてもよい。これらの他にも、種々の一致判定方法が可能である。

【0123】以下では、暗号化装置や復号装置の拡大鍵生成部の複数のラウンド処理部のバリエーションについて説明する。

【0124】図20には、ラウンド処理部の系列の構成例を示す。図20の例では、3段分示してあるが、各段の構成が所定段数従属接続されることになる。また、図20では、共通鍵が128ビットであり、各段の拡大鍵は64ビットである場合を例示している。図中、105は非線形写像Fであり、107の記号は排他的論理和を示している。

【0125】非線形写像Fは、全段で同一であってもよいし、段ごとに異なってもよい。また、後者の場合、基本的には、同じ構成を有するが、段に応じて異なる定数に依存するものであってもよい。

【0126】なお、図20と図20の逆関数のいずれを暗号化側（または復号側）に用いることも可能である。

【0127】一般に、差分解読法や線形解読法といった強力な解読法を用いても、最終段の拡大鍵のうち高々数ビットを特定するのが限界であるから、ラウンド関数の系列は、図20に示したような単純なFeistel構造でも安全性に大きな問題は無いと考えられるが、もし、より強力な解読法の出現に備えてより安全な構造を望むならば、例えば図21に示すようなラウンド関数の系列を用いてもよい。

【0128】図21の例では、2段分示してあるが、各段の構成が所定段数従属接続されることになる。また、図21では、共通鍵が128ビットであり、各段の拡大鍵は64ビットである場合を例示している。図中、109と111と113はそれぞれ非線形関数fとgとhを示しており、115の記号は排他的論理和を示している。非線形関数fとgとhは、すべて同一であっても、

すべて異なっても、それらのうち一部のもの同士のみ同一でもよい。

【0129】図22に、非線形関数fやgやhの構成例を示す。図22において、119は8ビットのS-boxであり、121はMDS行列に基づく32ビットの攪拌部である。

【0130】図21では、図20に比較して、出力される128ビットから中間状態を一意に決定することがより困難になっている。

10 【0131】図23に、図21の逆関数を示す。なお、図21と図23のいずれを暗号化側（または復号側）に用いることも可能である。

【0132】以下では、本発明を適用した暗号化装置の一具体例を示す。

【0133】図24に、本暗号化装置の構成例を示す。

【0134】この暗号化装置は、128ビット（64ビット）のブロック暗号であり、共通鍵は256ビット（128ビット）であり、1段の256ビット（128ビット）である場合を例にとっている。また、ラウンド関数の系列は、ラウンド・トリップ構成を有する場合を例にとっている。また、通常のSPN構造のS-boxの部分に小型のSPN構造を再帰的に埋め込んだ入れ子型SPN構造である場合を例にとっている。

【0135】図24において、データ攪拌部では、ラウンド関数(DU)201とラウンド関数(DD)203の繰り返し構造の後、ラウンド関数(DU)201とラウンド関数(DD(w o MDSH))205と、ラウンド関数(EXOR)207が接続されている。

30 【0136】また、拡大鍵生成部では、ユニット209とユニット211の対が1段分のラウンド関数に相当する。ただし、図24の例では、ユニット209とユニット209の間およびユニット211とユニット211の間が図1の中間状態ではなく、中間状態はユニット209やユニット211の内部に現れる構造になっている。

【0137】図25に、128ビットのブロック暗号の場合における、図24のユニット201の構成例を示す。図25において、215は鍵加算のための8ビットの排他的論理和であり、217は8ビットのS-boxであり、219はMDS行列に基づく32ビットの攪拌部である。213のユニットが4並列に設けられる。なお、64ビットのブロック暗号の場合には、213のユニットが2並列に設けられる。

【0138】図26に、128ビットのブロック暗号の場合における、図24のユニット202の構成例を示す。図26において、221は鍵加算のための8ビットの排他的論理和であり、223は16並列の8ビットのS-boxであり、225はMDS行列に基づく128ビットの攪拌部である。なお、64ビットのブロック暗号の場合には、223のS-boxが8並列に設けられる。

【0139】図27に、128ビットのブロック暗号の場合における、図24のユニット205の構成例を示す。図27において、227は鍵加算のための8ビットの排他的論理和であり、229は16並列の8ビットのS-boxである。なお、64ビットのブロック暗号の場合には、229のS-boxが8並列に設けられる。

【0140】ユニット207は、128ビットのブロック暗号の場合、ユニット205から出力される128ビットのブロック・データに128ビットの拡大鍵を加算するための排他的論理和である。また、ユニット207は、64ビットのブロック暗号の場合、ユニット205から出力される64ビットのブロック・データに64ビットの拡大鍵を加算するための排他的論理和である。

【0141】図28に、共通鍵のビット長が256ビットの場合における図24の拡大鍵生成部の構成例を示す。図28では、初段部分と折り返し部分のみにについて示している。図中、231は非線形関数Fであり、233は排他的論理和であり、235は段に依って異なる定数との排他的論理和である。237、239、241、243のユニットについては後述する。

【0142】図29に、共通鍵のビット長が128ビットの場合における図24の拡大鍵生成部の構成例を示す。図29では、初段部分と折り返し部分のみにについて示している。図中、251は非線形関数Fであり、253は排他的論理和であり、255は段に依って異なる定数との排他的論理和である。257、259、261、263のユニットについては後述する。

【0143】図30に、図28の非線形関数231の構成例を示す。図中、2311は排他的論理和であり、2313はS-boxであり、2315、2317については後述する。

【0144】次に、図28のP32と示されたユニット237、図29のP16と示されたユニット257、図30のP16と示されたユニット2315、図30のP8と示されたユニット2317について説明する。図31に、これらに共通する一般的な構成例を示す。図中、265は排他的論理和であり、あるiビットと他のiビットとの排他的論理和を取る操作が4回行われる。この構成をPiで表現したものが、各図のP8、P16、P32である。すなわち、図28のユニット237は図31の構成でi=32としたものであり、図29のユニット257は図31の構成でi=16としたものであり、図30のユニット2315は図31の構成でi=16としたものであり、図30のユニット2317は図31の構成でi=8としたものである。

【0145】図32に、図31のPiの逆変換であるPi⁻¹の構成例を示す。図中、267は排他的論理和である。図28のユニット243は図31の構成でi=32としたものであり、図29のユニット263は図31の構成でi=16としたものである。

【0146】なお、図30は、128ビットのブロック暗号の場合であったが、64ビットのブロック暗号の場合、すなわち図29の非線形関数251の場合には、図30において、P8をP4にし、P16をP8にすればよい。

【0147】次に、図28／図29の5と示されたユニット239／259、図29／図30のBと示されたユニット241／261について説明する。図33に、5と示されたユニットおよびBと示されたユニットの構成例を示す。両者の違いは、図33のユニット269における関数の内容である。

【0148】図33の構成は、ガロア体GF(2⁴)の元5、またはBを乗じるものである。

【0149】すなわち、入力となる32ビットを、4組の8ビットに分け、8ビット・データの同じ位置（例えば図33では最上位ビットの最下位ビットを例にとりて示している）の1ビットを集めて、これを1組4ビットのデータとし、8組の4ビット・データの各々を、GF(2⁴)の元とみなす。そして、各々の4ビット・データに対し各々のユニット269によって（ガロア体上の乗算に従って）5またはBを乗じた後に、各々のビットをそれぞれ対応するもとの位置に戻す。

【0150】なお、上記では、同じ位置のビットを取り出して処理を行うものとして説明したが、異なる位置のビットを（排他的に）取り出して処理を行うことも可能である。

【0151】ガロア体上の乗算は、表引きによっても、演算によっても、実回路によってもよい。

【0152】図34に、図33のユニット269の部分の構成例、すなわちGF(2⁴)上の乗算の結線表現（結線パターン）を、元5について（a）に、元Bについて（b）にそれぞれ示す。なお、前述したように、結合部分271では、排他的論理和がなされる。すなわち、この場合、図28／図29の5と示されたユニット239／259については、図33および図34（a）によって構成可能であり、図29／図30のBと示されたユニット241／261については、図33および図34（b）によって構成可能である。

【0153】ところで、データ攪拌部の拡大鍵を作用させる対象が既知あるいは比較的容易に推測可能な部分で使用される拡大鍵、例えばデータ攪拌部の最初の排他的論理和演算や、出力と鍵の推測からデータの推定が可能となる最後の鍵加算の前の排他的論理和演算部への拡大鍵において、異なる位置の演算要素単位（この例の場合、8ビット単位）で拡大鍵が常に一致することや常に一定の関係を持つことを防ぐようにすると好ましい。一構成例としては、上記のデータ攪拌部の最初の排他的論理和演算で使用される拡大鍵と、最後の鍵加算の前の排他的論理和演算部への拡大鍵に要素単位（この例の場合、8ビット単位）で常に成り立つ一致が発生しないよ

うに、拡大鍵を生成する（もしくは共通鍵を選択する）。これにより、サイドチャネル解析を容易ならしめる拡大鍵の一致や一定の関係の成立を妨げることができる。

【0154】なお、図24に対応する復号装置の構成は、データ攪拌部については図24のデータ攪拌部の逆関数になる。また、拡大鍵生成部については、暗号側と復号側ともに最終段のラウンド関数を備えた場合には、図24の拡大鍵生成部と同様の構成になる。もちろん、暗号側も復号側もそれぞれ最終段のラウンド関数を備えてもよい。

【0155】なお、図1～図34を参照しながら説明してきた以上のような実施形態において、128ビット等の特定のビット長を例にとったが、もちろん、どのようなビット長のブロックデータでも適用可能である。また、データ攪拌部は、どのような構成であっても適用可能である。

【0156】以下では、本実施形態のハードウェア構成、ソフトウェア構成について説明する。

【0157】本実施形態の暗号化装置や復号装置は、ハードウェアとしても、ソフトウェアとしても、実現可能である。

【0158】本実施形態は、ソフトウェアで実現する場合に、暗号化装置や復号装置を実現するプログラムであって、コンピュータに所定の手段を実行させるための（あるいはコンピュータを所定の手段として機能させるための、あるいはコンピュータに所定の機能を実現させるための）プログラムを記録したコンピュータ読取り可能な記録媒体としても実施することもできる。

【0159】また、ハードウェアとして構成する場合、半導体装置として形成することができる。

【0160】また、本発明を適用した暗号化装置や復号装置を構成する場合、あるいは暗号化プログラムや復号プログラムを作成する場合に、ブロックもしくはモジュールをすべて個別に作成することも可能であるが、同一構成を有するブロックもしくはモジュールについては1または適当数のみ用意しておいて、それをアルゴリズムの各部分で共有する（使い回す）ことも可能である。

【0161】また、ソフトウェアの場合には、マルチプロセッサを利用し、並列処理を行って、処理を高速化することも可能である。

【0162】なお、暗号化機能を持ち、復号機能を持たない装置として構成することも、復号機能を持ち、暗号化機能を持たない装置として構成することも、暗号化機能と復号機能の両方を持つ装置として構成することも、可能である。同様に、暗号化機能を持ち、復号機能を持たないプログラムとして構成することも、復号機能を持ち、暗号化機能を持たないプログラムとして構成することも、暗号化機能と復号機能の両方を持つプログラムとして構成することも、可能である。

【0163】次に、本実施形態のシステムへの応用について説明する。

【0164】本実施形態の暗号方式は、基本的にはどのようなシステムにも適用可能である。

【0165】例えば、図35に示すように、送信側装置301と、受信側装置303との間で、所定の方法もしくは手続により、鍵を安全に共有しておき、送信側装置301は送信データをブロック長ごとに本実施形態の暗号方式で暗号化し、所定のプロトコルに従って、通信ネットワーク302を介して、暗号文を受信側装置303へ送信し、暗号文を受信した受信側装置303では、受信した暗号文をブロック長ごとに本実施形態の暗号方式で復号し、もとの平文を得ることができる。なお、各々の装置が、暗号化機能と復号機能を両方持っていれば、双方向に暗号通信を行うことができる。

【0166】また、例えば、図36に示すように、計算機311では、所定の方法で鍵を生成し、保存したいデータをブロック長ごとに本実施形態の暗号方式で暗号化し、所定のネットワーク（例えば、LAN、インターネット等）314を介して、暗号化データとして、データ・サーバ313に保存しておく。計算機311では、このデータを読みみたいときは、データ・サーバ313から所望の暗号化データを読み込み、これをブロック長ごとに本実施形態の暗号方式で復号し、もとの平文を得ることができる。また、他の計算機312が、この鍵を知っていれば、同様に復号してもとの平文を得ることができるが、鍵の分からない他の計算機は、該暗号データを復号することはできず、情報のセキュリティ・コントロールが可能になる。

【0167】また、例えば、図37に示すように、コンテンツ提供側では、暗号化装置321により、あるコンテンツを、ある鍵で、ブロック長ごとに本実施形態の暗号方式で暗号化し、これを暗号化コンテンツとして、記録媒体322に記録し、これを頒布等する。記録媒体322を取得したユーザ側では、所定の方法で該ある鍵を入手することにより、復号装置323により、該コンテンツを、ブロック長ごとに本実施形態の暗号方式で復号し、コンテンツの閲覧もしくは再生等を行うことができる。

【0168】もちろん、上記以外にも種々のシステムに適用可能である。

【0169】なお、本実施形態で示した各々の構成は、一例であって、それ以外の構成を排除する趣旨のものではなく、例示した構成の一部を他のもので置き換えたり、例示した構成の一部を省いたり、例示した構成に別の機能を付加したり、それらを組み合わせたりすることなどによって得られる別の構成も可能である。また、例示した構成と論理的に等価な別の構成、例示した構成と論理的に等価な部分を含む別の構成、例示した構成の要部と論理的に等価な別の構成なども可能である。また、

例示した構成と同一もしくは類似の目的を達成する別の構成、例示した構成と同一もしくは類似の効果を奏する別の構成なども可能である。また、各種構成部分についての各種バリエーションは、適宜組み合わせることで実施することが可能である。また、本実施形態は、暗号化装置としての発明、復号化装置としての発明、システム全体としての発明、個別装置内部の構成部分についての発明、またはそれらに対応する方法の発明等、種々の観点、段階、概念またはカテゴリに係る発明を包含・内在するものである。従って、この発明の実施の形態に開示した内容からは、例示した構成に限定されることなく発明を抽出することができるものである。

【0170】本発明は、上述した実施の形態に限定されるものではなく、その技術的範囲において種々変形して実施することができる。

【0171】

【発明の効果】本発明によれば、拡大鍵生成のためのラウンド関数の系列を、共通鍵を入力し、共通鍵と同じ値を出力するように設定することによって、暗号化時と復号時の両方において、従来のような不必要な遅延時間や記憶容量の消費なく、共通鍵から *On-the-fly* に拡大鍵を生成することが可能になる。

【図面の簡単な説明】

【図1】本発明の一実施形態に係る暗号化装置の構成例を示す図

【図2】同実施形態に係る復号装置の構成例を示す図

【図3】ラウンド関数の系列の構成について説明するための図

【図4】ラウンド関数の系列の構成について説明するための図

【図5】ラウンド関数の系列の構成について説明するための図

【図6】ラウンド関数の系列の構成について説明するための図

【図7】ラウンド関数の系列の構成について説明するための図

【図8】ラウンド関数の系列の構成について説明するための図

【図9】ラウンド関数の系列の構成について説明するための図

【図10】同実施形態に係る暗号化装置の他の構成例を示す図

【図11】同実施形態に係る復号装置の他の構成例を示す図

【図12】拡大鍵と攪拌処理との接続関係例を示す図

【図13】拡大鍵と攪拌処理との接続関係例を示す図

【図14】拡大鍵と攪拌処理との接続関係例を示す図

【図15】拡大鍵と攪拌処理との接続関係例を示す図

【図16】同実施形態に係る暗号化装置のさらに他の構成例を示す図

【図17】同実施形態に係る復号装置のさらに他の構成例を示す図

【図18】同実施形態に係る暗号化装置のさらに他の構成例を示す図

【図19】同実施形態に係る拡大鍵生成部の構成例を示す図

【図20】同実施形態に係るラウンド処理部の構成例を示す図

10 【図21】同実施形態に係るラウンド処理部の他の構成例を示す図

【図22】図21のラウンド処理部の非線形関数ユニットの構成例を示す図

【図23】図21のラウンド処理部の逆関数を持つラウンド処理部の構成例を示す図

【図24】同実施形態に係る暗号化装置のさらに他の構成例を示す図

【図25】図24の第1のユニットDUの構成例を示す図

20 【図26】図24の第2のユニットDDの構成例を示す図

【図27】図24の第3のユニットDD (*w o M D S H*) の構成例を示す図

【図28】図24の拡大鍵生成部の構成例を示す図

【図29】図24の拡大鍵生成部の他の構成例を示す図

【図30】図28と図29の非線形関数ユニットを説明するための図

【図31】図28と図29の排他的論理和によるユニットを説明するための図

30 【図32】図28と図29の排他的論理和によるユニットを説明するための図

【図33】図28と図29のガロア体上の乗算によるユニットを説明するための図

【図34】図28と図29のガロア体上の乗算によるユニットを説明するための図

【図35】同実施形態の暗号方式を利用したシステムの一例を示す図

【図36】同実施形態の暗号方式を利用したシステムの他の例を示す図

40 【図37】同実施形態の暗号方式を利用したシステムのさらに他の例を示す図

【図38】従来の拡大鍵生成装置について説明するための図

【図39】従来の拡大鍵生成装置について説明するための図

【符号の説明】

1, 2…データ攪拌部

3, 4…拡大鍵生成部

11, 12…攪拌処理部

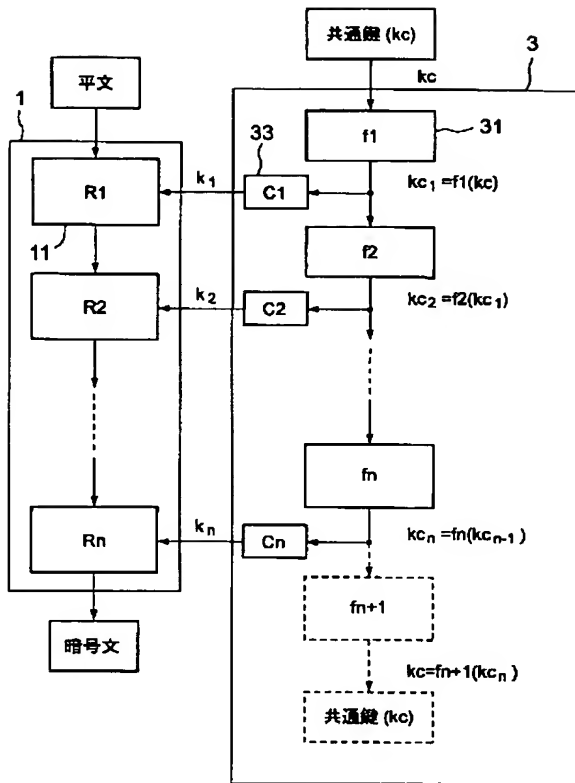
31, 42…ラウンド処理部

50 33, 44…拡大鍵変換部

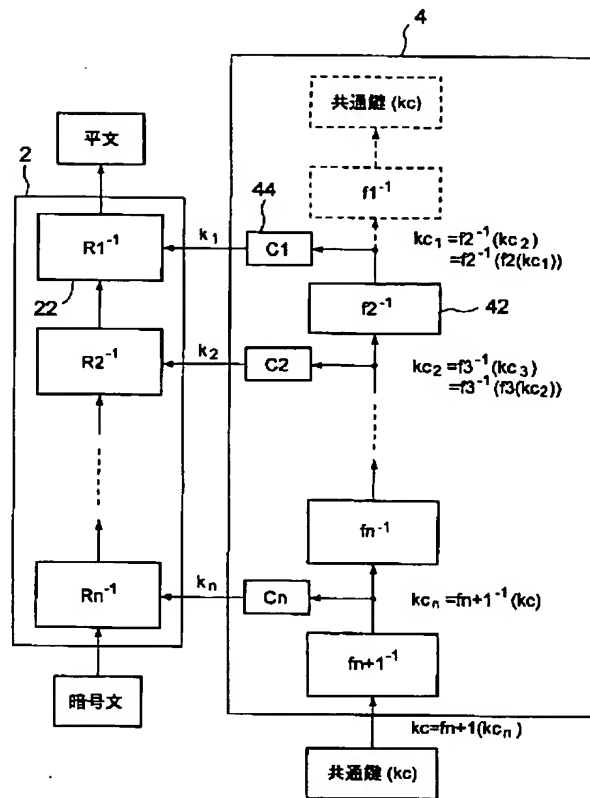
7, 8...デコーダ

15, 16...スイッチング回路

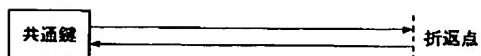
【図 1】



【図 2】



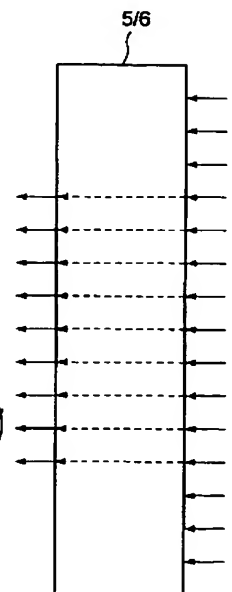
【図 3】



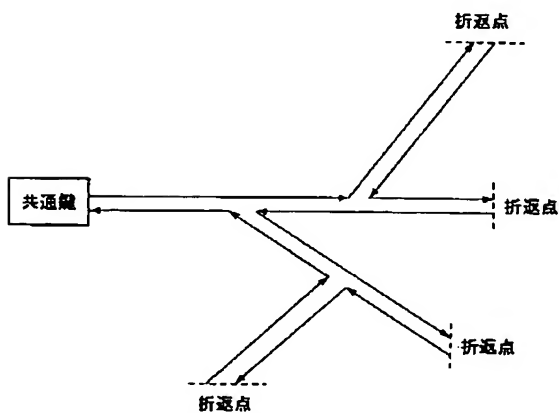
【図 4】



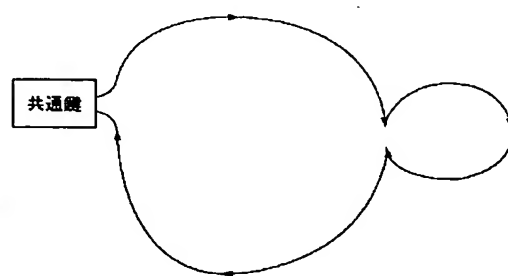
【図 13】



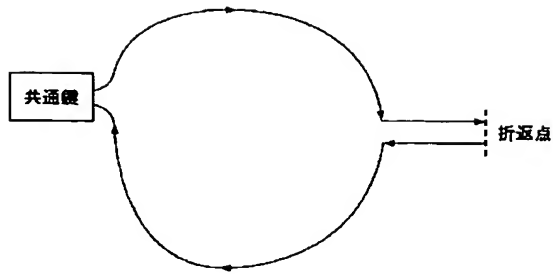
【図 5】



【図 6】



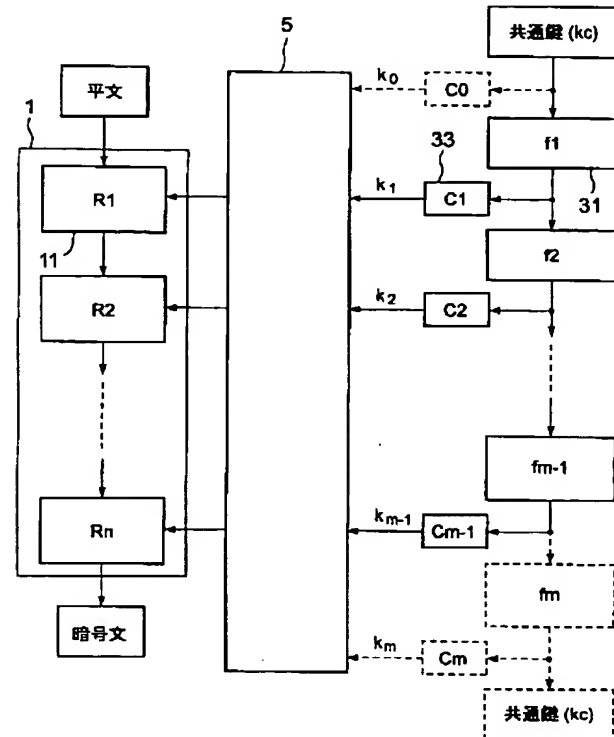
【図 7】



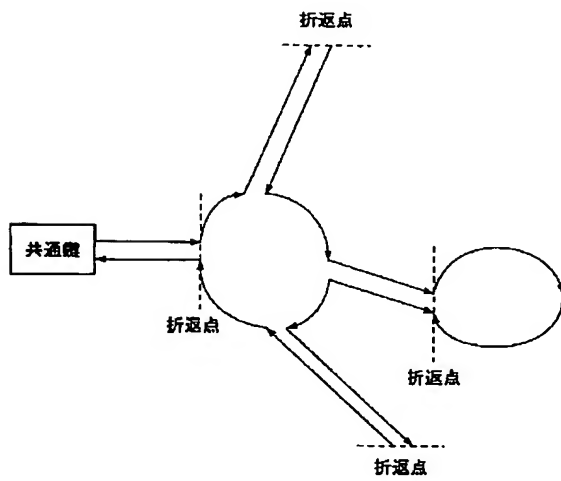
【図 8】



【図 10】

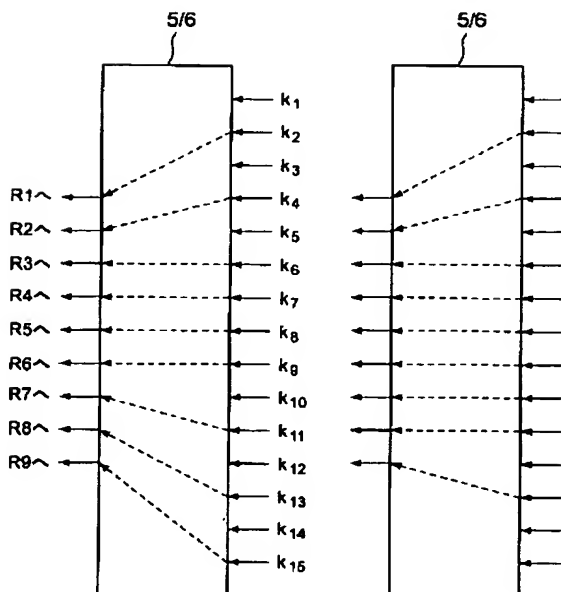


【図 9】

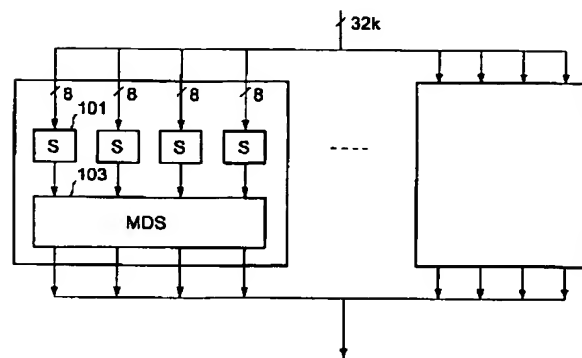


【図 12】

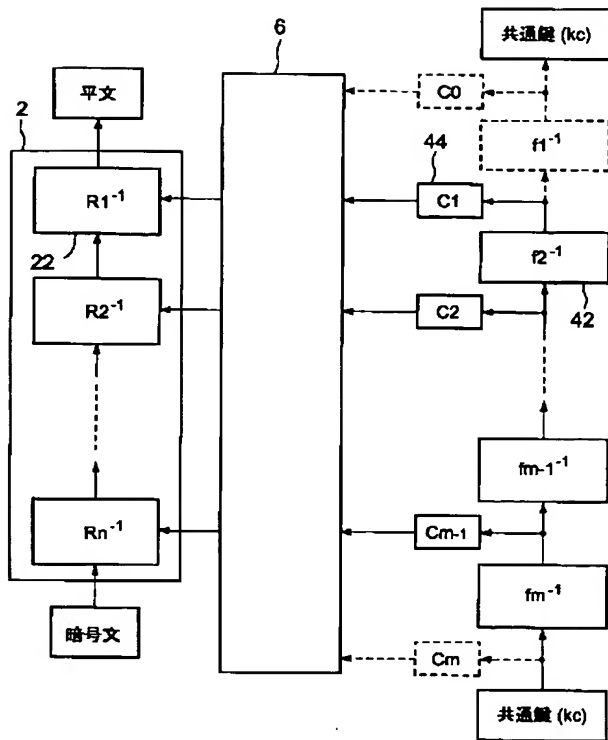
【図 14】



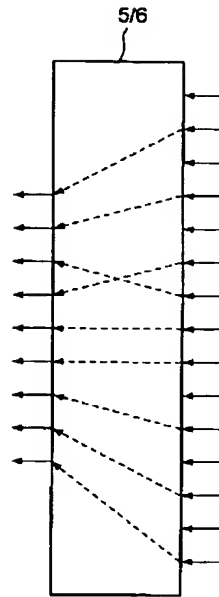
【図 19】



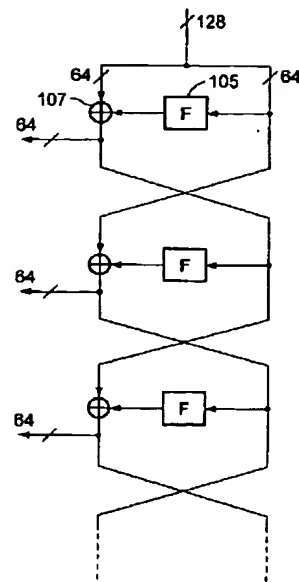
【図 11】



【図 15】

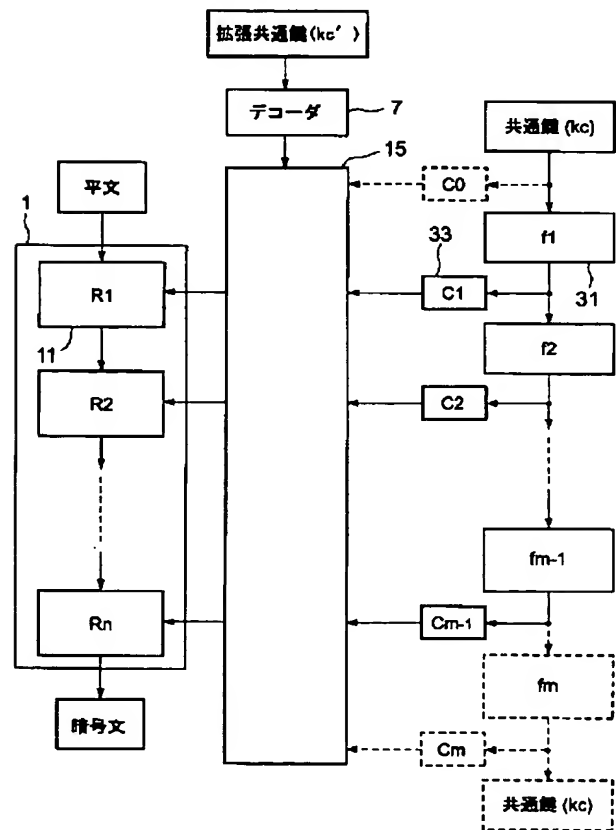
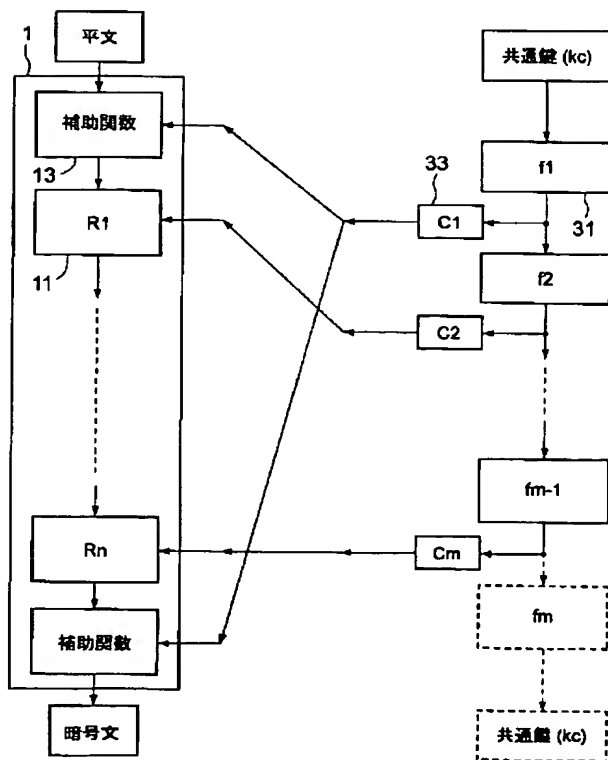


【図 20】

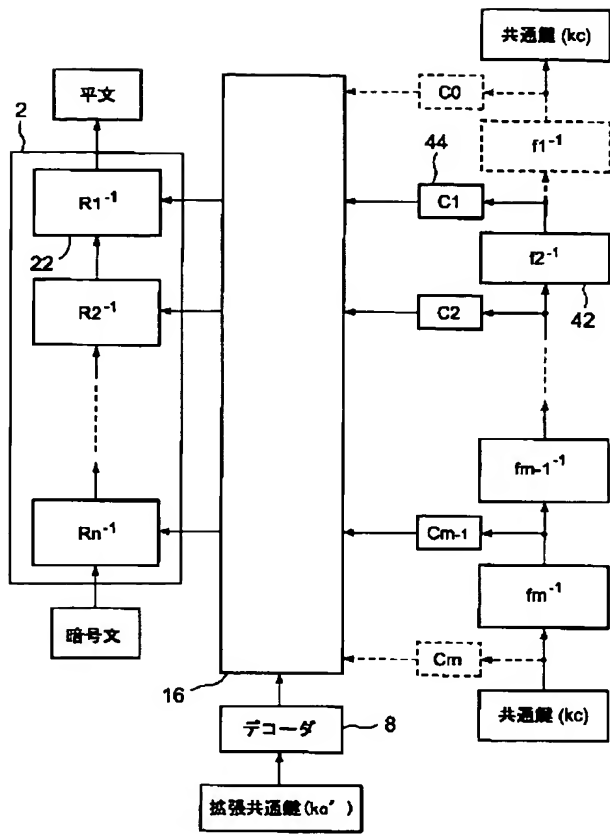


【図 16】

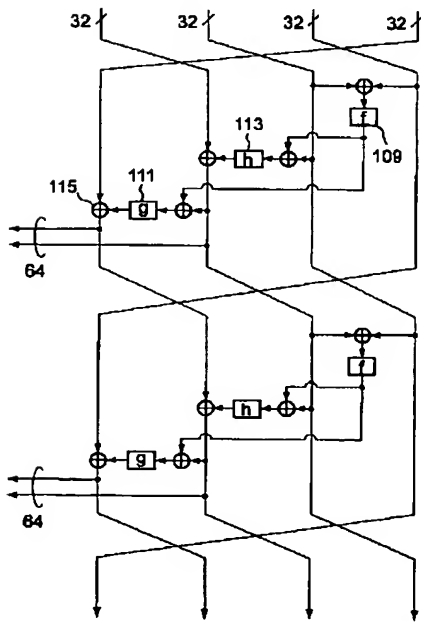
【図 18】



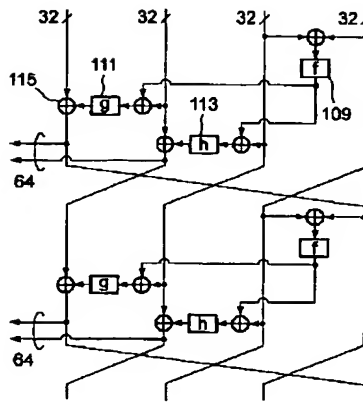
【図 17】



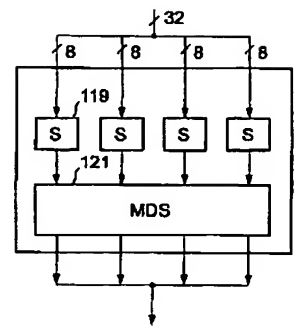
【図 23】



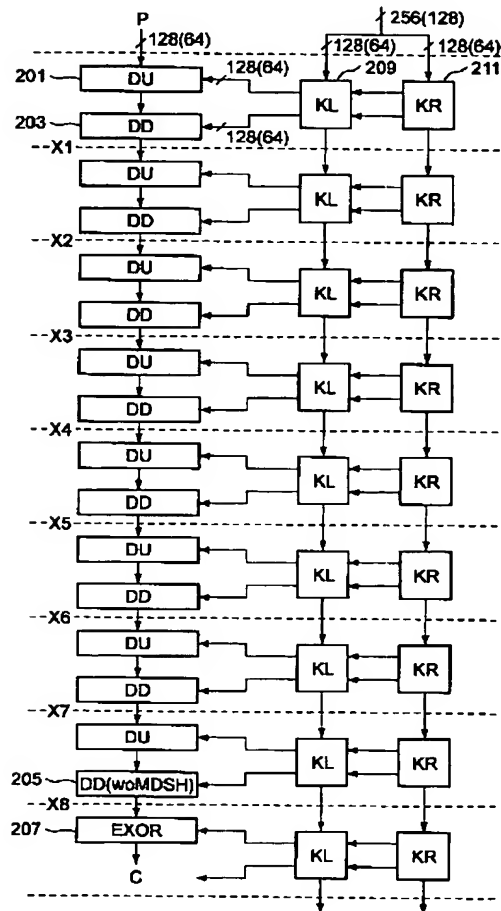
【図 21】



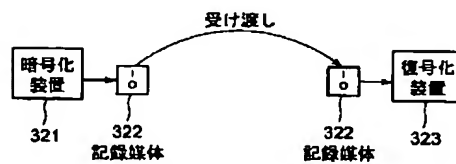
【図 22】



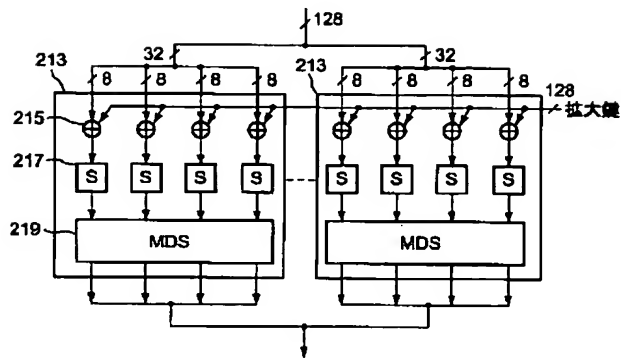
【図 24】



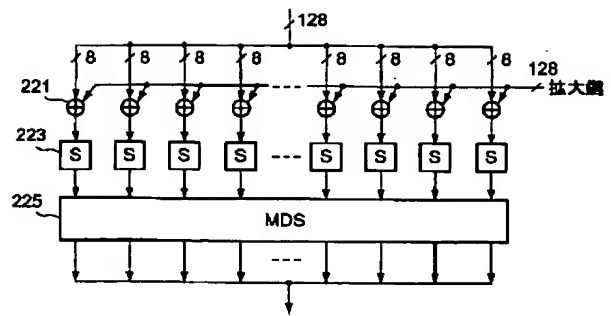
【図 37】



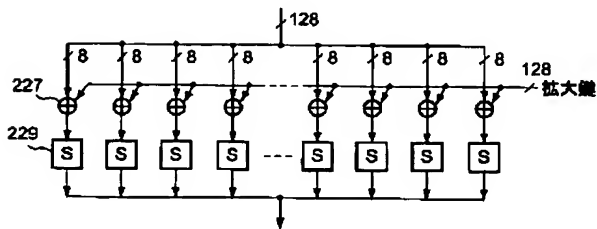
【図 25】



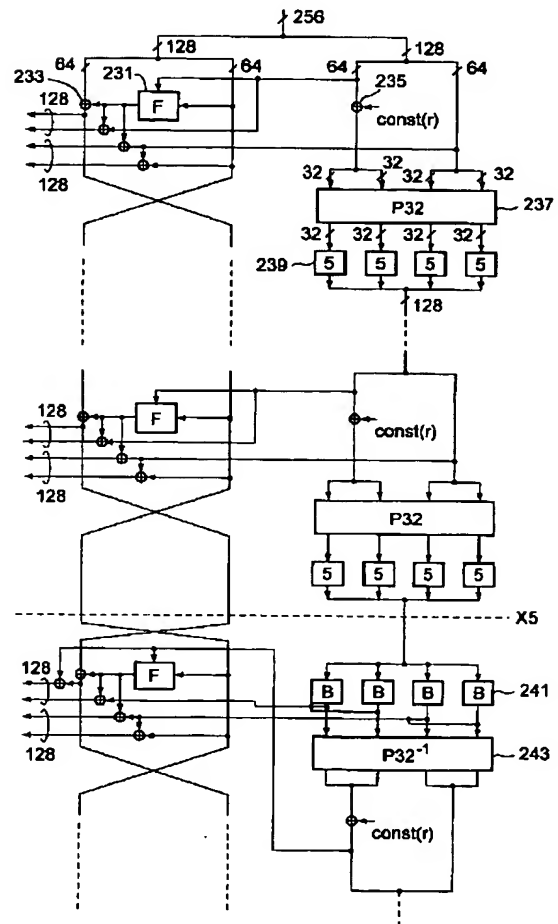
【図 26】



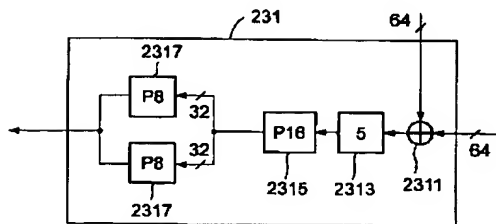
【図 27】



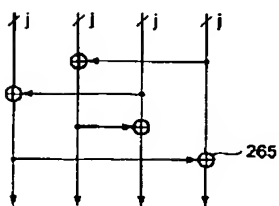
【図 28】



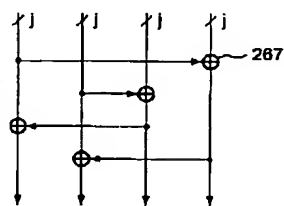
【図 30】



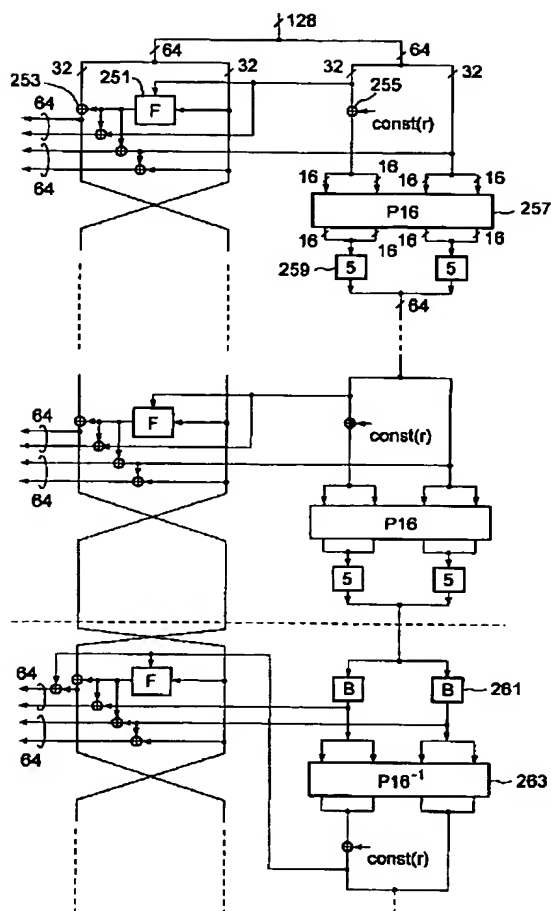
【図 31】



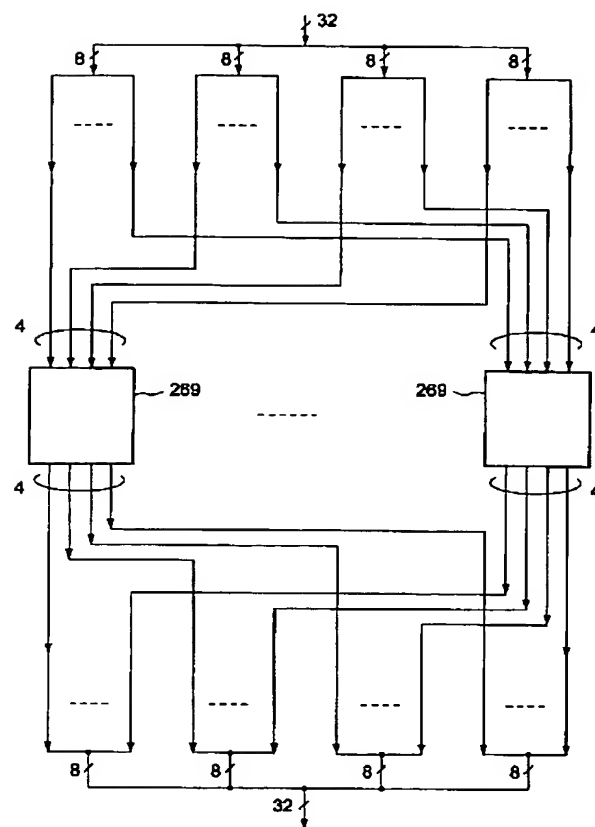
【図 32】



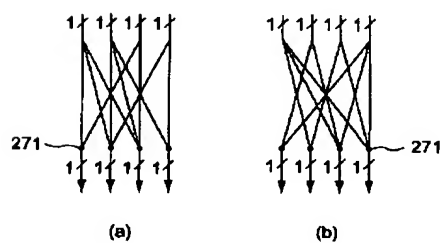
【図 29】



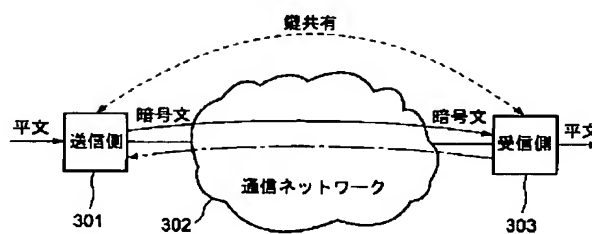
【図 33】



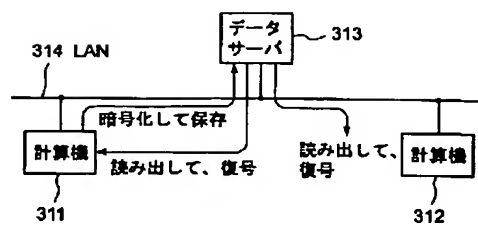
【図 34】



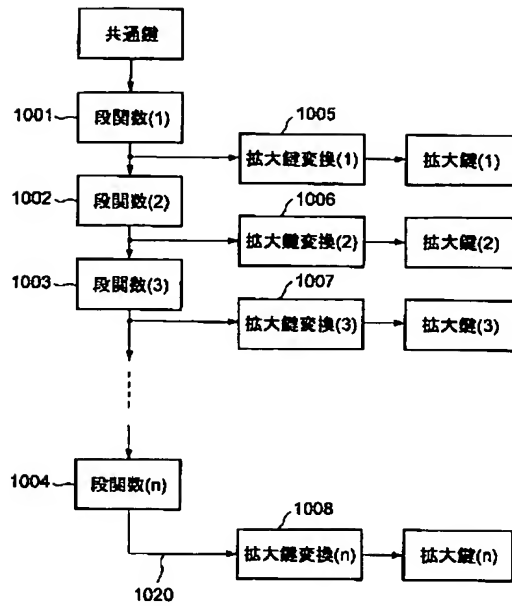
【図 35】



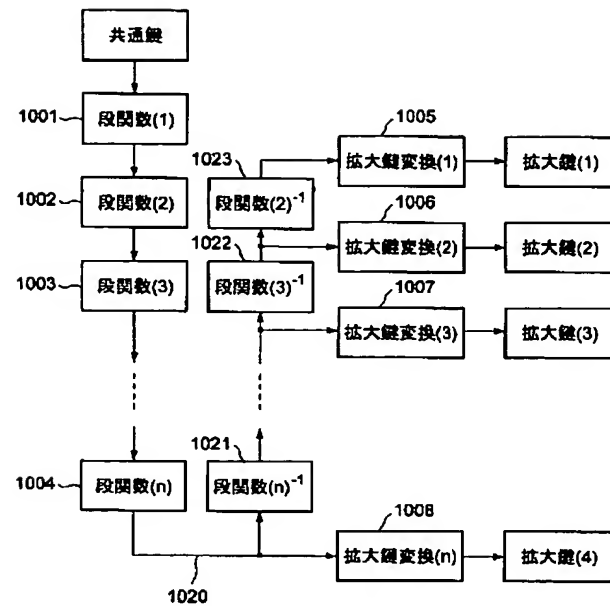
【図 36】



【図 38】



【図 39】



フロントページの続き

(72)発明者 大熊 建司
神奈川県川崎市幸区小向東芝町1番地 株
式会社東芝研究開発センター内

(72)発明者 佐野 文彦
東京都府中市東芝町1番地 株式会社東芝
府中事業所内

(72)発明者 川村 信一
神奈川県川崎市幸区小向東芝町1番地 株
式会社東芝研究開発センター内

Fターム(参考) 5J104 JA12 JA15 JA18 NA02 NA08